N93-25923

ADJUSTABLE LOSSLESS IMAGE COMPRESSION BASED ON A NATURAL SPLITTING OF AN IMAGE INTO DRAWING, SHADING, AND FINE-GRAINED COMPONENTS

Dmitry A. Novik
Mail Code 610.3
Universities Space Research Association
Greenbelt, Maryland 20771

James C.Tilton
Mail Code 930.4
NASA Goddard Space Flight Center
Greenbelt, Maryland 20771

Abstract. The compression, or efficient coding, of single band or multispectral still images is becoming an increasingly important topic. While lossy compression approaches can produce reconstructions that are visually close to the original, many scientific and engineering applications require exact (lossless) reconstructions. However, the most popular and efficient lossless compression techniques do not fully exploit the two-dimensional structural links existing in the image data. We describe here a general approach to lossless data compression that effectively exploits two-dimensional structural links of any length. After describing in detail two main variants on this scheme, we discuss experimental results.

1. Introduction

The problem of compressing or efficiently coding various types of data for transmission or archival has attracted much attention recently. Since the trend is continuing in the direction of collecting increasing amounts of space and Earth science data, this should be an important problem for some time to come.

Much of space and Earth science data is represented in image format. This is especially true for data that will be analyzed, recognized, and identified by human beings, with or without the assistance of a computer. The two-dimensional nature of single band images, and the three-dimensional nature of multispectral images, gives this type of data a different nature than one-dimensional data such as alphanumeric text.

Many different scientific and engineering applications utilize image archiving and/or transmission systems. Achieving the highest possible compression factors for the images will help reduce the expense of building and maintaining these systems. Besides space and Earth science applications, many other applications would benefit from compression, including medical and other scientific, industrial, defense, and intelligence image analysis applications; computer assisted image synthesis (e.g. animation); fingerprint and other criminology imaging application; and the publishing industry.

All of these applications require lossless compression of image data, especially in the associated archival systems. It should be emphasized that in many cases the absolute pixel values carry important information in addition to that carried by the general image morphology (as determined by pixel contrast). For example, in CAT or MRI medical images, the morphology represents the anatomy of the body, while the absolute pixel values carry additional information about pathology and disease.

The potential for compression in any meaningful two-dimensional still image data comes mainly from the statistical, spatial redundancy reflecting the two-dimensional structural links between image pixels that form two-dimensional structures. The more two-dimensional structure

contained in an image, the more compressible it will be, since this structure makes the image more dissimilar from a two-dimensional random field.

The amount of compression achieved for an image will depend on how deep and long are the structural links between image pixels, and how well this structure is extracted by the two-dimensional preprocessing of the image compression scheme. The most popular and efficient lossless compression techniques [1], [2], [3], do not effectively exploit these two-dimensional structural links, since they were designed for the compression of one-dimensional messages such as texts.

There are two well known approaches for exploiting the two-dimensional structure contained in still images. The first is two-dimensional predictive coding, in which the value of a pixel is predicted based on knowledge of the directly neighboring pixels previously scanned. Such preprocessing is recommended by JPEG Still Picture Compression Standard [4]. However, the restriction of using only directly neighboring pixels for prediction does not allow the use of any longer two-dimensional structural links between pixels.

The second well known approach for exploiting two-dimensional image structure is block encoding in which the image is artificially divided into k-by-k pixel blocks, and spatial features are extracted from the analysis of each of these blocks [4], [5], [6]. In real images, however, two-dimensional spatial structures generally do not correspond to such artificial blocks, but instead have arbitrary shapes and sizes.

2. General Approach

Our approach to two-dimensional image compression employs a natural splitting of an image into its drawing, shading and fine-grained components. This approach extracts and utilizes two-dimensional structural links of any length between image pixels. Because of this, our method has the potential of achieving better higher compression amounts than other methods that do not effectively utilize these two-dimensional links.

The splitting of an image into drawing, shading and fine-grained components arises naturally from the nature of digital images of natural scenes. A digital image is a matrix of $n_1 \times n_2 = N_0$ pixels, which often is the product of analog-to-digital conversion of a natural scene consisting of objects. Since the objects in the natural scene manifest themselves as connected regions in the two-dimensional scene, any so constructed digital image is a map-image of a number of connected regions that fully cover the image. We will call such an image a digital map-image.

A connected region is an ensemble of pixels of the same type or value, in which it is possible to travel from any pixel in such a region to any other pixel contained in it without intersecting the boundary of the region. For every meaningful digital map-image with N_0 pixels, the number of connected regions, N_{cr}, satisfies $N_{cr} << N_0$. This statistical redundancy is exploited by our approach.

In our approach, the drawing component is a contour map of boundaries between all N_{cr} connected regions of the digital map-image. (How these connected regions are defined is discussed in the next sections.) These boundaries may be represented by a "compound directional edgemap," which we will call a CDEdgemap. A CDEdgemap can be constructed by setting bits as follows:

i) set the first bit if (and only if) the pixel is in a different region from the pixel to the south;

ii) set the second bit if (and only if) the pixel is in a different region from the pixel to the east.

Relationships to pixels to the north and west can be found through reflection. Such two bit region boundary representations are sufficient to select and label separately all N_{cr} connected regions, with four nearest neighbor connectivity. This scheme can be extended to eight nearest neighbors by setting two more bits as follows:

iii) set the third bit if (and only if) the pixel is in a different region from the pixel to the southeast;
iv) set the fourth bit if (and only if) the pixel is in a different region from the pixel to the southwest.

Here the relationships to pixels to the northwest and northeast can be found through reflection.

The shading component is then the list of feature values for each region. The feature value is a common pixel value (or common pixel vector for the multispectral case) for each connected region. This common pixel value can be simply the region mean, or a more complicated compound value, such as a set of texture values. It is sufficient to list these feature values in the order in which the regions are first encountered in a fixed scan order through the image. Knowing the fixed scan order, the drawing and shading components can be combined to produce a possibly lossy reconstruction of the initial digital map-image. Depending on the parameter settings, this lossy reconstruction can be either identical or very similar to the original, or a coarse representation which may be appropriate for image browsing.

The fine-grained component is then the difference between original image and the lossy image reconstruction from the drawing and shading components. Since the drawing and shading components contain much of the structural information, the fine-grained component consists primarily of the region textures.

For further compression, the drawing, shading, and fine-grained components are then processed with a universal adaptive compression technique. We have used the LZW algorithm (available as "compress" on UNIX computers), and two variants of Arithmetic coding.

We have implemented two main variants of our general image compression approach. We will first discuss the variant based on dropping a specified number of least significant bits from the original image data. Later we will discuss the variant based on image segmentation by iterative parallel region growing.

3. Method Based on Dropping m_l Least Significant Bits

In the first variant, the drawing and shading components are defined by dropping a certain number of least significant bits from the original image. In this case, each connected region is the ensemble pixels of the same value (after a specified number of least significant bits are dropped), in which it is possible to travel from any pixel in such a region to any other pixel in the region without intersecting the of the boundary of the region.

Any image with m bits per pixel can be arbitrarily divided into two digital images. Each pixel of one image is the m_l least significant bits of the corresponding pixel in the original image. Each pixel of the other image is the m_h most significant bits of the corresponding pixel in the original image. Note that $m=m_h+m_l$. The splitting into drawing and shading components is performed only on the image with pixels corresponding to the m_h most significant bits in the original image.

The image with the m_l least significant bits is designated as the "fine-grained" component of the original digital image.

It is clear that, under our scheme, the size of the drawing and shading components will depend on the number of connected regions, N_{cr}. N_{cr} is determined by the nature of the original image, by the number of bits, m, in the analog-to-digital conversion used in creating the original image, and the number, m_l, of least significant bits dropped. N_{cr} will generally decrease with a decrease in m and increase in m_l.

Depending of the value of m_l, giving $m_h = m\text{-}m_l$, we will obtain different amounts of image compression. There will exist a value $m_{l,opt}$ that provides the maximum possible image data compression for the combination of all three components (drawing, shading and fine-grain) of a digital image represented with m bits per pixel.

The following is an outline of an algorithm that implements this variant of our image data compression scheme for an m-bit per pixel digital image:

i. select the m_l least important bits from original digital image as its fine-grain component;
ii. separate the image formed by the m_h most important bits into its shading and drawing components, producing a directional edge map and a feature list as described in the previous section;
iii. encode (compress) all three extracted image components by an appropriate universal lossless data compression technique.

The original image is reconstructed by following the steps listed below:

i. decode (decompress) the drawing and shading image components;
ii. construct a region label map by the following process: Give each image pixel the region label RL = (row-1)*ncol + col, where row is the row number, col is the column number, and ncol is the number of columns in the image. Then propagate the lowest region label through each region. Note that there is one unique location in each region where the final labeling matches the original labeling. These locations are also the locations of the first pixel encountered in a line-by-line scan of the image for each region. We call these locations the "seed" pixel for each region;
iii. while scanning through the region label map, fill in "seed" pixels in the reconstructed image from the shading component. Each time a new region is encountered, record the label in a table, and take the next value from the shading component file and place it in that location of the reconstructed image;
iv. propagate the seed pixel value to every pixel in each region using a connected component labeling algorithm;
v. decode (decompress) the fine-grained image component and add it to the reconstructed digital map-image formed by consolidation of the drawing and shading components.

4. Method Based on Image Segmentation

In the second variant of our compression method, the drawing and shading components are defined from an image segmentation produced by an iterative parallel region growing algorithm (IPRG), which is discussed in detail in [7] and [8]. Briefly, the IPRG algorithm is as follows:

i. Initialize the segmentation process by labeling each pixel as a separate region;
ii. merge all spatially adjacent pixels that have identical feature values;

iii. calculate a (dis)similarity criterion between each pair of spatially adjacent regions;
iv. merge pairs of regions that meet the merge constraints;
v. check for convergence. If converged, stop. Otherwise return to step iii.

A dissimilarity criterion is required in step iii. Dissimilarity criterion based on mean-square error and image entropy are described in [8]. For an image compression application, the most natural dissimilarity criterion is to minimize the range of pixel values in each region [9].

Let $MIN(D_k(p))$ be the minimum image value in region p for spectral band k, and let $MAX(D_k(p))$ be the maximum image value in region p for spectral band k.

The dissimilarity function for regions p and q in spectral band k is:

$$F_k(p,q) = MAX(\ MAX(\ D_k(p)\ ),MAX(\ D_k(q)\ )\ ) - MIN(\ MIN(\ D_k(p)\ ),MIN(\ D_k(q)\ )\ )$$

The dissimilarity function for regions p and q over all bands k, k=1...M is:

$$F(p,q) = \underset{k=1...M}{MAX}(F_k(p,q))$$

The merge constraint employed in step iv is that each merge must be the mutually best pairwise merge for merging regions. This is the same as merge constraint level 1.0 in [7].

Convergence in step iv occurs when there are no further merges that can be performed with the range of pixel values in each region remaining no more than $r_l = EXP(m_e * \ln(2)) - 1$, where m_e is the number of bits per pixel selected for the fine-grained component.

At convergence, the IPRG algorithm produces an edge map coded as described in the General Approach section above. This is the drawing component. It also produces the shading component as a list of feature values (or vectors for multispectral images), one per region, in the order each region is first encountered in a line-by-line scan through the image. The feature value is taken to be the "mid-val" of the regions, defined as the rounded average of the minimum and maximum pixel values in each region (the values are rounded *up* to the nearest integer, if necessary). The mid-val is taken as the feature value, rather than the region minimum or maximum, so that the image reconstructed from just the drawing and shading components will have close to the same average brightness as the original image.

The fine-grained component is defined in such a way that it will consist of positive integers in the range 0 to r_l. This is accomplished by taking the fine-grained component to be the difference between the original image and the image reconstructed from drawing and shading components, plus a bias value equal to $(r_l+1)/2$.

As it was in the first variant, in this second variant the size of the drawing and shading components will depend on the number of connected regions, N_{cr}, produced by the IPRG algorithm. As before, N_{cr} is determined by the nature of the original image, by the number of bits, m, in the analog-to-digital conversion used in creating the original image, and the number, m_e, of bits selected for the fine-grained component. N_{cr} will generally decrease with a decrease in m and increase in m_e.

Similar to the situation for the first variant, there will exist a value $m_{e,opt}$ that provides the maximum possible image data compression for the combination of all three components (drawing, shading and fine-grain) of a digital image represented with m bits per pixel.

Comparing the two variants, with $m_e=m_l$, the value of N_{cr} for the second variant will be no more than the value of N_{cr} for the first variant. Thus, the second variant will generally achieve better compression for the drawing and shading components. However, prior to extensive empirical tests, its is unclear as to which variant will produced more compressible fine-grained components. The second variant, however, requires significantly more computation for encoding than the first variant. However, the decoding process for both variants is virtually identical.

The following is an outline of an algorithm that implements this variant of our image data compression scheme for an m-bit per pixel digital image:

i. select value of m_e, or number of bits, for the fine-grain component;
ii. to help pack the feature values in the shading component in the fewest possible bits, find the minimum value of each spectral band and subtract for each pixel in each band;
iii. perform the IPRG algorithm on the adjusted image from step ii with convergence limit r_l = EXP(m_e*ln(2)) - 1. The IPRG algorithm produces the drawing and shading components as a directional edge map and feature list (of region mid-val's);
iv. compute the fine-grained component as the difference between the original image and the image constructed from the regions mid-val's, plus the bias value = $(r_l+1)/2$;
v. encode (compress) all three extracted image components by an appropriate universal lossless data compression technique.

The original image is reconstructed by following the same steps listed for image reconstruction for the first variant, with the exception that the bias value = $(r_l+1)/2$ must be subtracted and the band minimum value added back to each pixel in each band as the final step.

5. Experimental Results

Our compression approaches were tested on image data from:

-> the Landsat Thematic Mapper (TM) instrument (512 line by 512 column by 7 band section);
-> NOAA's Advance Very High Resolution Radiometer (AVHRR) instrument (Global Area Coverage) (512 line by 409 column by 5 band section);
-> JPL's Airborne Visible/Infrared Imaging Spectrometer (AVIRIS) (512 line by 614 column by 16 band section).

For both variants we found the optimal m_l or m_e (number of dropped bits or error bits), N_{CR} (number of connected regions), and CR (compression ratio defined as the number of bytes in the original image divided by the number of bytes in the compressed files). In addition, we compared our results to the compression produced by the Lempel-Ziv-Welch algorithm (UNIX "compress"), by two forms of arithmetic coding (Witten-Neal-Cleary and Adaptive), and by straightforward block averaging.

Our algorithm for straightforward block averaging is as follows:

i. select value of n_l, or size of blocks, for the spatial averaging;
ii. subtract the minimum value of each spectral band from each pixel in each band;
iii. compute region mid-value (average of region maximum and minimum) over n_l by n_l sized blocks and store as feature list;

iv. compute the fine-grained component as the difference between the original image and the image constructed from the blocks of mid-values, plus the bias value = $(r_l+1)/2$;*

v. encode (compress) all feature list and fine-grained image components by an appropriate universal lossless data compression technique.

The numerical results are presented in Tables 1, 2 and 3. The drawing, shading and fine-grained components were each losslessly compressed with one of the lossless techniques, depending on which technique performed best.

Table 1 shows that the natural splitting approaches did not perform very well with the TM image. An alternative view is that the adaptive arithmetic coder did so well it made our approaches look poor by comparison. In contrast, tables 2 and 3 show that our natural splitting approaches did very well compared to all other approaches. Our method based on Iterative Parallel Region Growing posted 47% and 60% improvements, respectively, over the Lempel-Ziv-Welch method on the AVHRR and AVIRIS image data. Similarly, the dropping m_l significant bits method showed gains of 41% and 42%, respectively over the Lempel-Ziv-Welch method on the same data sets. Smaller, but still significant, gains were obtained by both methods over the block averaging method.

The results obtained here encourage us to continue investigating the natural splitting approaches described herein. However, the relatively poor results on the TM image indicate that additional work may be required to truly exploit all the two-dimensional structural links that exist in the data.

References

[1] D. A. Huffman, "A Method for the Construction of Minimum Redundancy Codes," *Proceedings IRE*, Vol. 40, 1962, pp. 1098-1101.

[2] L. Ziv and A. Lempel, "A Universal Algorithm for Sequential Data Compression," *IEEE Trans. Information Theory*, Vol. 23, No. 3, 1977, pp. 337-343.

[3] I. H. Witten, *et al*, "Arithmetic Coding for Data Compression," *Communications of the ACM*, Vol. 30, No. 6, 1987, pp. 520-540.

[4] G. K. Wallace, "The JPEG Still Picture Compression Standard," *Communications of the ACM*, Vol. 34, No. 4, 1991, pp. 31-44.

[5] W. Chen and C. H. Smith, "Adaptive Coding of Monochrome and Color Images," *IEEE Trans. on Communications*, Vol. 25, No. 11, 1977, pp. 1285-1292.

[6] R. M. Gray, "Vector Quantization," *IEEE ASSP Magazine*, No. 4, 1984, pp. 4-28.

[7] J. C. Tilton, "Image Segmentation by Iterative Parallel Region Growing and Splitting," *Proceedings of the 1989 International Geoscience and Remote Sensing Symposium*, 1989, pp. 2420-2423.

[8] J. C. Tilton, "Experiences using TAE-Plus Command Language for an Image Segmentation Program Interface," *Proceedings of the Ninth TAE Users' Conference*, 1991.

[9] J. C. Tilton, "Hierarchical Data Compression: Integrated Browse, Moderate Loss, and Lossless Levels of Data Compression," *Proceedings of the 1990 International Geoscience and Remote Sensing Symposium*, 1990, pp. 1655-1658.

* r_l is computed by finding the maximum range of values in the difference between the original and reconstructed image, finding the number of error bits, n_e, required to represent this error and calculating $r_l = \mathrm{EXP}(n_e * \ln(2)) - 1$.

Table 1. Landsat Thematic Mapper (TM) Image

Washington, DC image; 7 spectral bands, 512 lines by 512 columns
1 byte pixels, 1,835,008 bytes in original

Method*	optimal n_i, m_l or m_e	N_{CR}	# bytes compressed	CR	$\frac{CR}{ADAP\text{-}CR}$	$\frac{CR}{BA\text{-}CR}$
LZW WNC ADAP	--- --- ---	--- --- ---	1,027,045 1,057,307 848,945	1.79 1.74 2.16	0.83 0.80 ---	0.88 0.86 1.07
BA	4	16,384	*(WNC: param.)* 905,017 *(ADAP: feat. & res.)*	2.03	0.94	---
DLSB	1	120,431 *(average)*	*(LZW: param.)* 952,580 *(ADAP: feat. & edge)* *(Bit-Pack: residual)*	1.93	0.89	0.95
IPRG	5	4,828	*(WNC: param. & feat.)* 890,743 *(ADAP: edge & res.)*	2.06	0.95	1.02

Table 2. Advanced Very High Resolution Radiometer (AVHRR) Image

5 spectral bands, 512 lines by 409 columns
2 byte pixels, 2,094,080 bytes in original

Method*	optimal n_i, m_l or m_e	N_{CR}	# bytes compressed	CR	$\frac{CR}{LZW\text{-}CR}$	$\frac{CR}{BA\text{-}CR}$
LZW WNC ADAP	--- --- ---	--- --- ---	1,121,165 1,229,692 1,292,551	1.87 1.70 1.62	--- 0.91 0.87	0.92 0.84 0.80
BA	8	3,328	*(WNC: param.)* 1,022,751 *(ADAP: feat.)* *(LZW: residual)*	2.04	1.09	---
DLSB	3	60,177 *(average)*	*(LZW: param.)* 795,258 *(ADAP: all others)*	2.63	1.41	1.29
IPRG	6	7.037	761,864 *(ADAP: all files)*	2.75	1.47	1.35

---

* LZW => Lempel-Ziv-Welch method, WNC => Witten-Neal-Cleary Arithmetic Coding, ADAP => Adaptive Arithmetic Coding, BP => Bit Packing, BA => Block averaging method, DLSB => Delete Least m_l Significant Bits method, IPRG => method based on Iterative Parallel Region Growing segmentation.

Table 3. Airborne Visible/Infrared Imaging Spectrometer Image

Moffet Field, CA image; 16 spectral bands (out of 210), 512 lines by 614 columns
10,059,776 bytes in original

Method*	optimal n_i, m_l or m_e	N_{CR}	# bytes compressed	CR	$\frac{CR}{LZW\text{-}CR}$	$\frac{CR}{BA\text{-}CR}$
LZW	---	---	6,799,419	1.48	---	0.89
WNC	---	---	6,883,323	1.46	0.99	0.88
ADAP	---	---	6,879,958	1.46	0.99	0.88
BA	4	19,712	*(WNC: param. & feat.)* 6,068,613 *(ADAP: residual)*	1.66	1.12	---
DLSB	5	71,666 *(average)*	*(ADAP or BP: residual)* 4,794,906 *(ADAP: all others)*	2.09	1.42	1.27
IPRG	7	16,267	*(WNC: param. & feat.)* 4,254,146 *(ADAP: residual)*	2.36	1.60	1.43

* LZW => Lempel-Ziv-Welch method, WNC => Witten-Neal-Cleary Arithmetic Coding, ADAP => Adaptive Arithmetic Coding, BP => Bit Packing, BA => Block averaging method, DLSB => Delete Least m_l Significant Bits method, IPRG => method based on Iterative Parallel Region Growing segmentation.